UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

MÉMOIRE D'ÉTUDIANT

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

CRT-595

LES ALGORITHMES DE RECUIT SIMULÉ EN OPTIMISATION COMBINATOIRE

par

René Séguin*

* Étudiant de Ph.D., département d'informatique et de recherche opérationnelle, Université de Montréal.

Université de Montréal
Centre de recherche sur les transports - Publication #595
Mai 1988

Dépôt légal - 3e trimestre 1988

RÉSUMÉ

Plusieurs problèmes d'optimisation combinatoire appartiennent à la classe des problèmes NP-complets; c'est-à-dire qu'il n'existe pas d'algorithmes qui produisent une solution exacte au problème en un temps polynomial (le temps de calcul augmente donc au moins exponentiellement avec la complexité du problème). Conséquemment, les solutions exactes pour les problèmes de grande dimension nécessitent des temps de calcul prohibitifs. On peut construire des algorithmes beaucoup moins exigeants en temps de calcul en appliquant des heuristiques conçues spécialement pour un problème particulier et qui s'efforcent de trouver une solution quasi optimale. Cependant, la plupart du temps ces algorithmes dépendent fortement de la structure du problème considéré. Ceci est un inconvénient majeur puisqu'il empêche toute application ou implantation rapide et flexible. De plus, il existe un nombre croissant de problèmes combinatoires originant de domaines les plus divers pour lesquels il n'y a pas de méthode d'optimisation adéquate connue (par exemple, dans le domaine de la conception d'ordinateurs). C'est pour ces raisons qu'on recherche une méthode d'optimisation flexible et généralement applicable. Le recuit simulé peut être vu comme une de ces méthodes : c'est une technique d'optimisation générale pour résoudre les problèmes combinatoires.

Le but de ce travail est de donner un aperçu de ce qu'est le recuit simulé et de son application aux problèmes d'optimisation combinatoire. Le texte débute par une section introduisant l'optimisation combinatoire et l'amélioration itérative (un algorithme classique pour ces problèmes), puis on montre l'analogie qui existe entre l'optimisation combinatoire et certains phénomènes physiques. Par la suite, on introduit le recuit simulé et ses différentes composantes; on décrit son fonctionnement et on étudie sa convergence. Finalement, on montre comment appliquer la méthode de recuit simulé aux principaux problèmes d'optimisation combinatoire.

Mots-clés : algorithmes; heuristiques; optimisation combinatoire; recuit simulé; système physique;

1) Introduction.

Plusieurs problèmes d'optimisation combinatoire appartiennent à la classe des problèmes NP-complets; c'est-à-dire qu'il n'existe pas d'algorithmes qui produisent une solution exacte au problème en un temps polynomial (le temps de calcul augmente donc au moins exponentiellement avec la complexité du problème) [26]. Conséquemment, les solutions exactes pour les problèmes de grande dimension nécessitent des temps de calcul prohibitifs. On peut construire des algorithmes beaucoup moins exigeants en temps de calcul en appliquant des heuristiques conçues spécialement pour un problème particulier et qui s'efforcent de trouver une solution quasi optimale. Cependant, la plupart du temps ces algorithmes dépendent fortement de la structure du problème considéré. Ceci est un inconvénient majeur puisqu'il empêche toute application ou implantation rapide et flexible. De plus, il existe un nombre croissant de problèmes combinatoires originant de domaines les plus divers pour lesquels il n'y a pas de méthode d'optimisation adéquate connue (par exemple, dans le domaine de la conception d'ordinateurs). C'est pour ces raisons qu'on recherche une méthode d'optimisation flexible et généralement applicable. Le recuit simulé peut être vu comme une de ces méthodes: c'est une technique d'optimisation générale pour résoudre les problèmes combinatoires.

La technique provient de méthodes de simulation par ordinateur utilisées pour des sytèmes physiques, et fut employée pour la première fois avec succès pour un problème d'optimisation combinatoire par Kirkpatrick et al. [18] . À partir de ce moment, la méthode fut reconnue et appliquée à plusieurs problèmes dans diverses disciplines [10] . Le travail de Kirkpatrick est basé essentiellement sur l'analogie entre un système physique constitué de particules en interaction et un problème d'optimisation combinatoire de grande taille. La résolution du problème d'optimisation suppose en effet la

minimisation d'une fonction de coût; cet objectif peut être comparé à la recherche d'un état d'énergie minimale pour un ensemble de particules. Pour conduire un sytème physique dans un état de basse énergie, on utilise la technique du recuit qui consiste à augmenter la température jusqu'à ce que les particules soient distribuées aléatoirement dans la phase liquide du système, puis on abaisse lentement la température (pour éviter les structures qui correspondent à des minima locaux de l'énergie) pour faire évoluer le système vers son équilibre thermodynamique. À haute température tous les états peuvent être atteints, mais les états de faible énergie sont plus probables; lorsqu'on diminue la température, le nombre d'états accessibles décroît et le système atteint finalement son équilibre (un état solide stable). Ceci est valable si la température de départ est assez élevée et le refroidissement suffisamment lent. En optimisation combinatoire, une situation similaire se produit: le système peut prendre plusieurs configurations différentes et chacune a un coût associé. La transposition des algorithmes utilisés en physique à un problème d'optimisation suppose que l'on introduira, pour celui-ci, un paramètre de contrôle analogue à la température.

Le but de ce travail est de donner un aperçu de ce qu'est le recuit simulé et de son application aux problèmes d'optimisation combinatoire. Le texte débute par une section introduisant l'optimisation combinatoire et l'amélioration itérative (un algorithme classique pour ces problèmes), puis on montre l'analogie qui existe entre l'optimisation combinatoire et certains phénomènes physiques. Par la suite, on introduit le recuit simulé et ses différentes composantes; on décrit son fonctionnement et on étudie sa convergence. Finalement, on montre comment appliquer la méthode de recuit simulé aux principaux problèmes d'optimisation combinatoire.

Mentionnons, en terminant, que ce travail est inspiré en majeure partie de la thèse de doctorat de Siarry [30] et de deux articles de Aarts et van Laarhoven [1, 3].

2) Problème combinatoire et amélioration itérative.

Un problème d'optimisation combinatoire peut être caractérisé par une fonction objective (ou fonction de coût) que l'on cherche à optimiser et un ensemble de contraintes qui définissent l'ensemble de configurations possibles (solutions réalisables). En pratique, le nombre de variables et de contraintes peut être très élevé et la fonction de coût doit prendre en considération de nombreux objectifs plus ou moins contradictoires. Elle est donc en quelque sorte un indicateur de la "qualité globale" d'une configuration.

On peut représenter un problème d'optimisation combinatoire à l'aide de l'ensemble R des configurations r (r est un vecteur qui détermine uniquement une configuration), et par une fonction de coût $C : R \rightarrow \mathbb{R}$, qui assigne un nombre réel $C(r)$ à chaque configuration r. R est déterminé par l'ensemble fini des configurations possibles, donc par les contraintes. Nous allons assumer que C est défini de telle sorte que plus sa valeur est petite, meilleure est la configuration correspondante. L'objectif du problème d'optimisation est donc de trouver une configuration r_o telle que

$$C(r_o) = C_{opt} = min\{C(r) : r \in R\}, \tag{1}$$

où C_{opt} dénote le coût optimal.

L'amélioration itérative est un algorithme classique pour les problèmes d'optimisation combinatoire dont le principe est le suivant: on part d'une configuration initiale, qui peut être choisie complètement au hasard ou bien par un utilisateur. On effectue alors un réarrangement (par exemple, en permuttant deux composantes du vecteur r), et on compare les valeurs de la fonction de coût avant et après cette modification du vecteur. La différence de coût, si on passe de r_i à r_j, est donnée par:

$$\Delta C_{ij} = C(r_j) - C(r_i). \tag{2}$$

Si le changement conduit à une diminution de coût, ($\Delta C_{ij} < 0$) il est accepté, et la configuration obtenue (r_j) sert de point de départ pour un nouvel essai. Dans le cas contraire ($\Delta C_{ij} > 0$), on revient à la configuration précédente (r_i) avant de faire une autre tentative de réarrangement. Le processus est itéré jusqu'à ce qu'on ne puisse plus améliorer la solution.

La limitation inhérente à cet algorithme, lorsqu'on emploie des réarrangements relativement simples, est qu'il peut demeurer coincé dans un minimum local. Pour éviter ce problème, on doit utiliser certaines astuces pour être assuré que la solution finale est à proximité d'un minimum global. Si l'algorithme est coincé dans un minimum local, on peut alors utiliser des réarrangements plus complexes; mais le choix de ceux-ci est fortement relié à la nature du problème et requiert une très bonne connaissance du domaine particulier d'application. De plus, plusieurs problèmes d'optimisation combinatoire nécessitent une implantation rapide et flexible. Les méthodes d'optimisation, impliquant des réarrangements complexes et des manipulations dépendant de problèmes particuliers, ne sont donc pas appropriées.

3) Optimisation combinatoire et phénomènes physiques.

La structure complexe de l'espace des configurations a inspiré une analogie avec des phénomènes physiques, qui a conduit à une nouvelle méthode itérative, dite méthode du recuit simulé qui évite les minima locaux. Nous allons examiner maintenant cette analogie et, pour mieux saisir celle-ci, nous utiliserons un problème particulier, soit l'archétype des problèmes combinatoires: le problème du voyageur de commerce (PVC). Ce problème est le suivant: on doit visiter N villes, une et une seule fois, revenir au point de départ et ce en minimisant la distance parcourue.

Pour illustrer l'analogie, supposons qu'un planificateur peu qualifié relie les villes qui vont se suivre dans l'itinéraire, en choisissant toujours la ville la plus éloignée; il est clair que la route obtenue est inacceptable (voir Figure 1). Le rôle d'un programme pour résoudre le PVC est de transformer cette situation désordonnée pour aboutir à un itinéraire ordonné où les villes se succèdent de manière à minimiser la distance. En d'autres termes, ce programme doit réaliser une transformation désordre-ordre, qui, partant d'un "liquide de villes", aboutit à un "solide" ordonné. Or, une telle transformation se produit spontanément dans la nature si l'on abaisse progressivement la température d'un système.

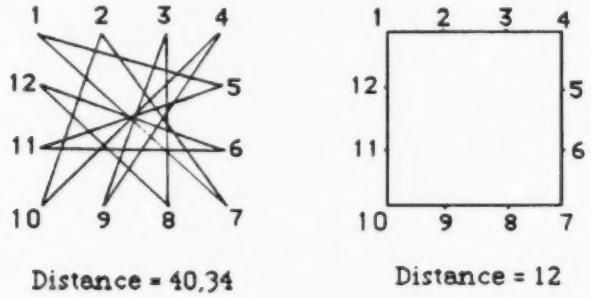


Figure 1 - Transformation désordre-ordre réalisée par un programme de voyageur de commerce. (La distance entre les villes i et i + i (i = 1,...,11) et entre les villes 12 et 1 est de une unité.)

Pour modifier l'état d'un système ou d'un matériau, le physicien dispose souvent d'un paramètre de commande qu'il peut fixer: la température. Le recuit est précisément une stratégie de contrôle de la température en vue d'approcher un état optimum (basse énergie). Cette technique consiste à chauffer préalablement le matériau jusqu'à le porter à l'état liquide (qui correspond à une énergie élevée). Puis on refroidit très lentement le matériau, en marquant des paliers de température de durée suffisante; si la descente en température est trop rapide, il apparaît des défauts qui peuvent être éliminés par réchauffement local. Cette stratégie de baisse contrôlée de la température conduit à un état solide stable, correspondant à un minimum absolu de l'énergie. La technique opposée est celle de la trempe, qui consiste à

abaisser très rapidement la température du matériau en fusion: on obtient dans ce cas un état métastable qui correspond à un minimum local de l'énergie. Avec la technique du recuit, le refroidissement du matériau en fusion a provoqué une transformation désordre-ordre, tandis que la technique de la trempe a abouti à figer un état désordonné.

La présence de nombreux minima locaux dans un problème tel que celui de la Figure 1 est facile à justifier. Puisque le critère à minimiser est la longueur de l'itinéraire, la distance, pour aller d'une ville à la suivante sur l'itinéraire, aura tendance à être aussi courte que possible; cependant, si on déplace une ville dans l'itinéraire, on va raccourcir certains trajets entre villes successives mais on va également en allonger d'autres. Un minimum global résultera donc d'un compromis, entre des exigences conflictuelles, qui engendrera des "frustrations". Cette même notion de frustration est apparue en physique, pour la première fois, dans le domaine des verres de spins [30]. Un verre de spins est un alliage magnétique dans lequel chaque atome possède soit un moment magnétique "positif", soit un moment magnétique "négatif". Ces spins contribuent à l'énergie totale par interactions deux à deux. (voir Binder et Young [6]).

Il existe des techniques numériques qui permettent de simuler, sur ordinateur, le comportement, en fonction de la température, d'ensembles de particules en interaction ; il était donc très tentant de transposer ces techniques aux problèmes d'optimisation combinatoire. L'idée d'utiliser la technique du recuit, en vue de traiter ces problèmes, a donné naissance à la méthode du recuit simulé. Elle consiste à introduire, en optimisation, un paramètre de contrôle, qui joue le rôle de la température. La "température" du système à optimiser doit avoir le même effet que la température du système physique: elle doit conditionner le nombre d'états accessibles et conduire vers l'état optimal si elle est abaissée de façon lente et bien contrôlée, et vers un minimum local si elle est abaissée brutalement.

L'algorithme de Metropolis [24] , qui permet de simuler le comportement d'un système physique en équilibre thermodynamique à une température T donnée, consiste à engendrer une séquence de configurations qui forme une chaîne de Markov, chaque configuration étant déduite de la précédente conformément à la règle suivante: considérons une configuration r d'énergie (C_r); pour trouver la configuration r + 1, nous mettons en oeuvre la procédure suivante:

i - on perturbe légèrement la configuration r ; par exemple, dans le PVC, on intervertit l'ordre de visite de deux villes successives; puis on calcule la nouvelle énergie C(r + 1);

ii - si $C(r+1) \leq C(r)$, on accepte cette nouvelle configuration: on incrémente l'indice r, puis on va à l'étape i ;

iii - si $C(r + 1) > C(r)$, on calcule $A = exp((C(r) - C(r + 1)) / T)$, et on tire au hasard un nombre p compris entre 0 et 1; si $p \leq A$, on accepte la nouvelle configuration: on incrémente r, et on va à l'étape i; si $p > A$, on refuse cette configuration et on va à l'étape i.

Le paramètre de contrôle T est la température. À très haute température, tous les changements sont acceptés: l'algorithme équivaut à une marche aléatoire dans l'espace des configurations. Dans la prochaine section, nous allons voir plus en détail cet algorithme ainsi que la façon de l'appliquer au cas des problèmes combinatoires.

4) Recuit simulé.

Comme pour l'algorithme d'amélioration itérative, l'algorithme de recuit simulé débute avec une configuration initiale et génère ensuite une suite de nouvelles configurations à l'aide de réarrangements. Ici, une règle

d'acceptation entre en jeu; elle nous indique s'il faut accepter ou refuser un réarrangement. Cette règle ne permet pas que des améliorations mais aussi des détériorations. La règle d'acceptation est construite de telle sorte qu'initialement, les détériorations sont acceptées avec de fortes probabilités. Au fur et à mesure que l'algorithme progresse, la règle d'acceptation est modifiée de manière à réduire de plus en plus la probabilité d'accepter des détériorations et cela jusqu'à ce qu'elle devienne presque nulle à la fin de l'algorithme. De cette façon, on protège l'algorithme de la possibilité de rester coincé dans un minimum local et on lui permet d'atteindre un minimum global en n'utilisant que des réarrangements simples.

Plusieurs éléments interviennent dans le fonctionnement de l'algorithme: la structure de l'espace des configurations (reliée au choix des réarrangements), la règle d'acceptation et le programme de recuit (il détermine de quelle façon modifier la règle d'acceptation). Nous allons voir en détails chacun de ces éléments.

4.1 Espace des configurations.

L'espace des configurations joue un rôle fondamental dans l'efficacité de la méthode du recuit simulé pour résoudre un problème complexe d'optimisation. Pour comprendre cette importance, revenons au cas du PVC à N villes. Puisque, dans ce problème, la ville de départ et le sens de la tournée n'ont aucune importance, il existe (N-1)!/2 tournées différentes. Cet ensemble des configurations possibles devient l'espace des configurations à condition d'être doté d'une "topologie" . Celle-ci découle directement du fait que l'on se propose de résoudre le problème par une méthode itérative: il faut définir les réarrangements possibles à partir d'une configuration donnée. Par exemple, dans le cas du PVC, on peut adopter la procédure suivante: les tournées voisines d'une tournée donnée sont obtenues en inversant deux villes dans l'itinéraire (par exemple, l'itinéraire A-B-C-D-E pourrait devenir A-B-D-C-E). Lorsque deux configurations se correspondent par un changement élémentaire, elles peuvent être considérées comme voisines; la topologie résulte immédiatement de cette notion de proximité entre deux

configurations: la "distance", entre deux configurations, est définie comme le nombre minimum de changements élémentaires nécessaires pour passer d'une configuration à l'autre. En outre, à chaque configuration correspond une énergie (la longueur de la tournée dans le cas du PVC), si bien que l'espace des configurations est caractérisé par un paysage d'énergie. Toute la difficulté du problème d'optimisation réside dans le fait que ce paysage comporte un grand nombre de vallées plus ou moins profondes et plus ou moins proches, qui correspondent à des minima locaux de l'énergie. Un problème apparemment délicat à traiter peut être grandement simplifié, soit par un élargissement de l'espace des configurations, qui déstabilise les minima locaux, sans en créer de nouveaux, soit par le choix d'une meilleure topologie. Nous allons illustrer chacune de ces possibilités par un exemple.

Pour comprendre l'effet de l'élargissement de l'espace des configurations, nous utiliserons le problème du placement de composants électroniques sur une carte: il s'agit de placer des composants de dimensions très diverses en minimisant la surface occupée. Pour traiter ce problème, on part d'une configuration initiale, que l'on modifie itérativement en autorisant deux sortes de changements: les translations d'un composant ou les permutations de deux composants. Un changement peut provoquer le chevauchement de composants, ce qui est interdit dans la solution recherchée. Il pourrait, cependant, être avantageux de permettre cette situation de manière transitoire au cours du processus d'optimisation. Néanmoins, pour terminer avec une solution réalisable (sans chevauchement), il faut pénaliser cette situation à l'aide d'un terme spécifique dans la fonction de coût (de manière aussi à ne pas créer de nouveaux minima locaux). Cet exemple montre l'avantage que l'on peut retirer de l'élargissement de l'espace des configurations permises.

Voyons maintenant l'avantage que peut apporter un meilleur choix de la topologie. Considérons, à cet effet, l'exemple du PVC de la Figure 2 et supposons ensuite, que les seuls réarrangements permis sont les permutations de deux villes qui se suivent dans l'itinéraire. Si on applique la méthode d'amélioration itérative classique à l'itinéraire de départ A-B-C-D-E-F, aucune

amélioration n'est effectuée parce que chaque permutation augmente le coût du trajet. Néanmoins, si nous avions aussi permis les changements de position d'une ville dans l'itinéraire, nous aurions trouvé la solution A-B-D-E-C-F en ne modifiant que la position de la ville C. Cette solution étant meilleure que la précédente, on voit que l'inclusion du dernier type de réarrangement permet à l'algorithme classique de trouver la solution optimale et donc que la topologie choisie affecte grandement les performances d'un algorithme.

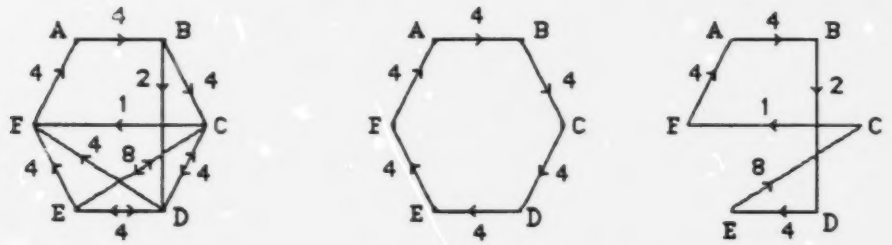


Figure 2.- Exemple de PVC montrant l'importance des réarrangements autorisés.

La convergence de l'algorithme du recuit simulé, que nous étudierons en détails dans les sections suivantes, ne dépend pas de la topologie de l'espace des configurations, mais celle-ci peut avoir un effet déterminant sur la vitesse de convergence: Siarry [30] mentionne que la méthode du recuit semble procurer d'excellents résultats en un temps raisonnable lorsque la topologie est choisie de façon telle que la méthode déterministe obtienne elle-même, et fréquemment, d'assez bonnes solutions.

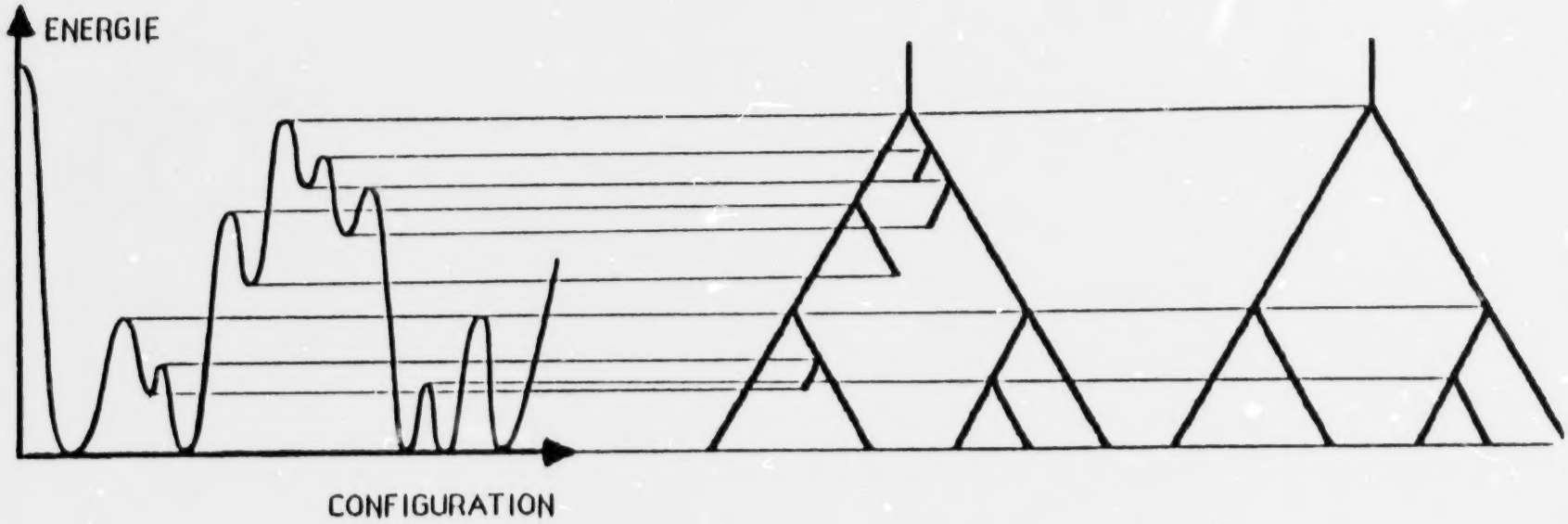
Kirkpatrick et al. [18] ont mis l'accent sur l'analogie entre le paysage d'énergie qui caractérise l'espace des configurations d'un verre de spins et celui qui est obtenu dans le cas d'un problème d'optimisation combinatoire. Il est donc naturel d'essayer de faire bénéficier le domaine de l'optimisation des progrès accomplis dans celui des verres de spins. Siarry [30] affirme qu'il a été démontré que la distribution des "vallées" (minima) dans l'espace des configurations, pour le cas des verres de spins, possède une structure

ultramétrique. Nous allons donner quelques détails au sujet de l'ultramétricité et voir de quelle façon elle est liée au recuit simulé.

Un espace métrique est un espace doté d'une distance d , qui satisfait l'inégalité triangulaire: étant donnés trois points A, B et C, on a: $d(A,C) \leq d(A,B) + d(B,C)$. Un espace ultramétrique est un espace métrique qui satisfait l'inégalité suivante, qui est plus forte: $d(A,C) \leq \text{Max}[d(A,B), d(B,C)]$. L'inégalité ultramétrique implique que, dans un espace ultramétrique, tous les triangles sont, soit équilatéraux, soit isocèles à base étroite. Dans un ensemble ultramétrique, il n'y a pas de point intermédiaire entre deux points donnés. Voyons maintenant le lien entre l'ultramétricité et le recuit simulé.

Revenons au paysage d'énergie qui caractérise un problème d'optimisation complexe; le relief tourmenté de ce paysage peut être représenté à l'aide d'un arbre des bifurcations (voir Figure 3). La grandeur associée à la dimension verticale de l'arbre est l'énergie; les noeuds (c'est-à-dire les points de bifurcation) et les feuilles (les extrémités des branches) de l'arbre correspondent respectivement aux maxima et minima locaux du paysage d'énergie (ou de sa coupe dans un plan). On considérera l'arbre ultramétrique associé (Figure 3), qui est obtenu en éliminant toutes les branches dont l'extrémité n'est pas située sur la ligne horizontale de base (les minima qui ne sont pas globaux).

Rammal et al. [27] affirment qu'il est possible de définir, pour tout paysage d'énergie, une distance ultramétrique entre deux minima locaux: c'est la valeur de la barrière d'énergie qu'il faut franchir pour passer de l'un à l'autre. C'est ce que l'on retrouve dans l'arbre ultramétrique associé de la Figure 3. Solla et al. [31] suggèrent l'hypothèse que le recuit simulé serait plus efficace pour traiter les problèmes d'optimisation pour lesquels il existe une forte corrélation entre la distance naturelle (le nombre de réarrangements nécessaires pour passer d'un point à un autre) et la distance ultramétrique que l'on vient juste de définir. Une telle corrélation nous indiquerait que les



paysage d'énergie (ou sa coupe dans un plan) "arbre des bifurcations" associé "arbre ultramétrique" associé

Figure 3 - Construction de l'arbre des bifurcations et de l'arbre ultramétrique.

vallées larges sont profondes et que les vallées étroites le sont beaucoup moins. Donc, lorsqu'il y a une forte corrélation, l'algorithme du recuit aurait tendance à explorer les vallées profondes, puisqu'il est facile de sortir des vallées moins profondes, celles-ci étant moins larges donc nécessitant peu de réarrangements. Une corrélation parfaite est indiquée lorsqu'on peut interchanger, sur l'axe vertical, l'énergie et le nombre de réarrangements et ce pour l'arbre ultramétrique associé. Évidemment, la corrélation n'a pas à être absolument parfaite pour appliquer le résultat. On peut mentionner, en terminant cette section, que les approches du recuit simulé fondées sur l'analyse des paysages d'énergie sont encore à un stade exploratoire.

4.2 Programme de recuit et règle d'acceptation.

Nous allons nous intéresser maintenant au programme de recuit qui contrôle la décroissance des probabilités d'accepter des détériorations et ce, à l'aide d'un paramètre qu'on appellera "température". Le contrôle est exercé de façon à atteindre le plus vite possible une bonne solution. Durant le processus, il est possible de réduire ces probabilités, après chaque réarrangement. Ici, on va plutôt utiliser l'approche pour laquelle les probabilités, entre chaque diminution, sont gardées inchangées pour un certain nombre de réarrangements. L'avantage de cette approche est que le processus d'optimisation peut alors être décrit comme une suite finie de chaînes de Markov homogènes, pour lesquelles le comportement à la limite peut plus facilement être décrit que dans le cas non homogène.

On rappelle que R est l'espace des configurations et que r est un "vecteur d'état" dont les composantes définissent entièrement une configuration donnée (ou un "état"). On considère de plus l'ensemble I_R des numéros attribués à chacune des configurations de R:

$$I_R = \{1, 2, \ldots, |R|\}, \tag{3}$$

en notant $|R|$ le cardinal de R. On rapelle aussi que $C(r_i)$ est la valeur de la fonction de coût en l'état i, de vecteur d'état r_i .

Une chaîne de Markov est une suite d'états pour lesquels la probabilité d'apparition est déterminée par la probabilité de transition de l'état i à l'état j définie, dans notre cas, par

$$T_{ij}(\beta) = B_{ij}(\beta) P_{ij} \qquad \text{si} \qquad i \neq j \qquad (4)$$

et

$$T_{ij}(\beta) = 1 - \sum_{k \neq i} B_{ik}(\beta) P_{ik} \qquad \text{si} \qquad i = j \qquad (5)$$

où $B_{ij}(\beta)$ dénote la probabilité d'acceptation, c'est-à-dire la probabilité d'accepter l'état j lorsqu'on se trouve dans l'état i, à la température β ; P_{ij} dénote la probabilité de pertubation, c'est-à-dire la probabilité d'engendrer l'état j lorsqu'on se trouve dans l'état i. Lorsque la température β est maintenue constante, la probabilité de transition $T_{ij}(\beta)$ est stationnaire, et la chaîne de Markov correspondante est dite homogène. On peut ajouter que la probabilité de transition $T_{ij}(\beta)$ ne dépend que des états i et j et pas des états antérieurs à i ; en d'autres termes, tout le passé du système est résumé par l'état courant.

Pour chaque état i, le sous-ensemble R_i est formé de toutes les configurations qui peuvent être obtenues en un seul réarrangement à partir de l'état i. De plus, I_{R_i} est l'ensemble des numéros de ces configurations. Le système est généralement perturbé en choisissant au hasard un réarrangement parmi ceux qui sont permis. Conséquemment, les probabilités de pertubation P_{ij} sont distribuées uniformément dans l'espace des configurations R :

$$P_{ij} = \begin{cases} |R_i|^{-1} & \text{si} \quad j \in I_{R_i} \\ 0 & \text{si} \quad j \notin I_{R_i} \end{cases} . \tag{6}$$

L'algorithme du recuit simulé trouve un minimum global si après un nombre (possiblement élevé) de transitions, disons K, la relation suivante est vérifiée:

$$\mathbb{P}\{X(K) \in R_o\} = 1, \tag{7}$$

où $X(K)$ est le résultat de la K^{e} transition et R_o, l'ensemble des configurations minimales globales. Pour que la chaîne de Markov homogène (et donc l'algorithme) converge, il faut qu'il existe, sous certaines conditions sur les probabilités $B_{ij}(\beta)$ et P_{ij}, un vecteur d'équilibre (ou distribution stationnaire) donné par le $|R|$-vecteur $q(\beta)$ et défini comme suit:

$$q^t(\beta) = \lim_{k \to \infty} a^t T^k(\beta) \tag{8}$$

où a dénote la distribution de probabilité initiale des configurations, k est le nombre de transitions et $T(\beta)$, la matrice de transition associée à $T_{ij}(\beta)$. De plus, le vecteur d'équilibre doit converger, lorsque β décroît vers 0, vers une distribution uniforme sur l'ensemble des configurations optimales:

$$\lim_{\beta \downarrow 0} q(\beta) = \pi \tag{9}$$

où les composantes du $|R|$-vecteur π sont données par

$$\pi_i = \begin{cases} |R_o|^{-1} & \text{si} \quad i \in R_o \\ 0 & \text{sinon} \end{cases} . \tag{10}$$

Aarts et van Laarhoven [1] notent que lorsqu'on utilise la méthode du recuit simulé, on peut imposer les conditions suivantes:

$$\text{(i)}\quad \forall i,j\in I_R\ ,\ \exists K_{ij}\in \mathbb{N}\ ,\ \left\{k_1,k_2,\ldots,k_{K_{ij}}\right\}\subset I_R:\quad \text{(connexité)}\qquad (11)$$

$$k_1=i\ ,\quad k_{K_{ij}}=j\quad \text{et}\quad T_{k_\ell\, k_{\ell+1}}\neq 0\ ,\ \forall \ell\in\{1,\ldots,K_{ij}-1\}$$

$$\text{(ii)}\quad \forall i,j\in I_R\ ,\quad P_{ij}=P_{ji}\qquad \text{(réversibilité)}\qquad (12)$$

$$\text{(iii)}\quad \forall i,j\in I_R\ ,\quad B_{ij}(\beta)=1\qquad \text{si}\qquad \Delta C_{ij}\leq 0\qquad (13)$$

$$\text{(iv)}\quad \forall i,j\in I_R\ ,\quad B_{ij}(\beta)\in(0,1)\subset\mathbb{R}\qquad \text{si}\qquad \Delta C_{ij}>0\qquad (14)$$

avec les conditions spéciales

$$\lim_{\beta\to\infty} B_{ij}(\beta)=1\qquad (\Delta C_{ij}>0)\qquad (14a)$$

et

$$\lim_{\beta\downarrow 0} B_{ij}(\beta)=0\qquad (\Delta C_{ij}>0)\qquad (14b)$$

$$\text{(v)}\quad \forall i,j,k\in I_R \text{ avec } C(r_k)\geq C(r_j)\geq C(r_i),\ B_{ik}(\beta)=B_{ij}(\beta)B_{jk}(\beta)\qquad (15)$$

La condition (i) nous assure que deux états quelconques, i et j , se correspondent par un nombre fini de réarrangements; la condition (iii), que tous les réarrangements, qui n'augmentent pas l'énergie, sont acceptés; la condition (iv), que tous les réarrangements, qui augmentent l'énergie, sont acceptés à température infinie, refusés à température nulle et acceptés avec une probabilité non nulle aux températures intermédiaires. Il existe un ensemble de conditions plus générales et moins restrictives, mais beaucoup moins utilisées (Romeo et al. [28]).

En utilisant les résultats concernant les chaînes de Markov homogènes [17, 29], on peut établir les propriétés suivantes:

Propriété 1:

Considérons le processus de Markov engendré par un mécanisme de transition qui respecte les cinq conditions énoncées plus haut; ce mécanisme est appliqué k fois, à température β constante, à partir d'une configuration initiale donnée i. La distribution des fréquences relatives correspondant à cette configuration est $(a_{ij} = \delta_{ij}$, le delta de Kronecker); donc la configuration i a une probabilité de 1 et toutes les autres configurations ont une probabilité nulle. Lorsque k tend vers l'infini, la chaîne de Markov obtenue possède le vecteur d'équilibre $\mathbf{q}(\beta)$, qui est indépendant de la configuration initiale. La i-ème composante de ce vecteur, soit $q_i(\beta)$, représente la probabilité que le système se trouve dans la configuration i lorsque le régime stationnaire est atteint.

Propriété 2:

Les composantes du vecteur d'équilibre sont données par:

$$q_i(\beta) = \frac{B_{i_0 j}(\beta)}{\sum_{i=1}^{|R|} B_{i_0 j}(\beta)} \qquad (16)$$

où i_0 désigne le numéro d'une configuration optimale.

Propriété 3:

Lorsque la température tend vers l'infini ou vers zéro, les valeurs limites de $q_i(\beta)$ sont données par:

$$\lim_{\beta \to \infty} q_i(\beta) = |R|^{-1} \qquad (17)$$

et

$$\lim_{\beta \downarrow 0} q_i(\beta) = \begin{cases} |R_o|^{-1} & \text{si } i \in I_{R_o} \\ 0 & \text{si } i \notin I_{R_o} \end{cases} \quad \text{(c'est la distribution } \pi \text{ ; voir (10))} \qquad (18)$$

où R_o, l'ensemble des configurations optimales est donné par:

$$R_o = \{ r_i \in R \mid C(r_i) = C(r_{i_0}) \} \qquad (19)$$

La propriété 3 se déduit immédiatement de la propriété 2 en utilisant la condition (iv) (Éq. (14) à (14b)); son interprétation est la suivante: pour les grandes valeurs de la température, toutes les configurations peuvent être obtenues avec la même probabilité; en revanche, lorsque la température tend vers zéro, le système atteint une configuration optimale avec une probabilité égale à l'unité.

Nous allons maintenant étudier la règle d'acceptation $B_{ij}(\beta)$: nous verrons les effets, sur cette règle, des conditions que nous avons imposées, puis nous ferons d'autres choix pour arriver à une règle bien déterminée. Ceci nous permettra d'obtenir une expression particulière pour le vecteur d'équilibre de l'équation (16).

En utilisant les conditions (iii) et (v)(Éq. (13) et (15)), Aarts et van Laarhoven [1] montrent que pour une valeur donnée de β , la fonction $B_{ij}(\beta)$, ne dépend que de $C(r_i)$ et $C(r_j)$. On peut donc factoriser la fonction $B_{ij}(\beta)$, et, en utilisant la condition (iv) (Éq. (14), on obtient:

$$\forall \Delta C_{ij} > 0 \quad , \quad B_{ij}(\beta) = \exp(- g(C(r_i), C(r_j)) \, \alpha(\beta)) \qquad (20)$$

où $g: \mathbb{R}^2 \to \mathbb{R}^+$ et $\alpha : \mathbb{R}^+ \to \mathbb{R}^+$. On peut alors réécrire la condition (v) (Éq. (15)) en fonction de la fonction g. On obtient après quelques manipulations:

$$\forall i, j, k \in I_R \text{ avec } C(r_k) \geq C(r_j) \geq C(r_i), \qquad (21)$$

$$g(C(r_i), C(r_k)) = g(C(r_i), C(r_j)) + g(C(r_j), C(r_k)).$$

Aarts et van Laarhoven [1] font l'hypothèse supplémentaire suivante: la fonction g satisfait la condition qui suit:

$$\forall x, y, z \in [a, b] \subset \mathbb{R} \quad \text{avec} \quad x \geq y \geq z, \qquad g(x,z) = g(x,y) + g(y,z). \tag{22}$$

Ces mêmes auteurs affirment que cette condition, même si elle est plus forte que celle de l'équation (21), n'est pas trop restrictive car on peut supposer que $C(r_i)$ est distribué de façon raisonnablement uniforme, puisque l'espace des configurations est habituellement très large. Ils ajoutent, de plus, qu'on peut montrer que la fonction g peut prendre la forme suivante:

$$g(x, y) = h(y) - h(x) \qquad \forall x, y \in [a, b]. \tag{23}$$

On a donc:

$$\forall i, j \in I_R \quad \text{et} \quad \Delta C_{ij} > 0 \quad g(C(r_i), C(r_j)) = h(C(r_j)) - h(C(r_i)) \tag{24}$$

où h: $\mathbb{R} \rightarrow \mathbb{R}^+$ est monotone croissante. Il y a plusieurs fonctions qui satisfont cette équation. On choisit ici:

$$h(C(r_i)) = C(r_i). \tag{25}$$

Maintenant qu'on connaît la fonction g, nous allons étudier la fonction $\alpha(\beta)$ pour laquelle nous avons, en utilisant les équations (14a), (14b) et (20),

$$\lim_{\beta \rightarrow \infty} \alpha(\beta) = 0 \quad \text{et} \quad \lim_{\beta \downarrow 0} \alpha(\beta) = \infty. \tag{26}$$

Ici aussi, plusieurs fonctions sont possibles, et nous choisissons

$$\alpha(\beta) = \beta^{-1}. \tag{27}$$

En utilisant les équations (13), (20), (24), (25) et (27), on trouve finalement, comme expression de la règle d'acception:

$$B_{ij}(\beta) = \begin{cases} \exp\left(-\Delta C_{ij} / \beta\right) & \text{si} \quad \Delta C_{ij} > 0 \\ 1 & \text{si} \quad \Delta C_{ij} \leq 0 \end{cases}, \tag{28}$$

et conséquemment, les fréquences relatives de la distribution stationnaire sont données par:

$$q_i(\beta) = \frac{\exp\left(-\Delta C_{i_0 i} / \beta\right)}{\sum_{j=1}^{|R|} \exp\left(-\Delta C_{i_0 j} / \beta\right)}. \tag{29}$$

L'expression qu'on retrouve à l'équation (28) est la règle d'acceptation standard pour l'algorithme du recuit simulé et elle est équivalente au critère de Metropolis que nous avons introduit à la fin de la section 3. Cette règle est couramment utilisée du fait qu'elle découle directement de l'analogie entre les simulations en physique statistique et l'optimisation d'un problème combinatoire de grande taille. La distribution stationnaire de l'équation (29) est connue sous le nom de distribution de Boltzmann. Il est possible de construire des variantes à cette règle: en effet, nous aurions pu faire d'autres choix que ceux des équations (25) et (27). De plus, Siarry [30] mentionne que rien ne nous assure que la règle standard est la plus efficace du point de vue temps de calcul.

En pratique, on obtient une chaîne de Markov en générant de façon répétitive une nouvelle configuration à partir d'une ancienne et en appliquant, pour chaque nouvelle, la règle d'acceptation de l'équation (28) pour une valeur donnée de β. Le processus d'optimisation est effectué en commençant la génération des chaînes avec une certaine valeur de β et en répétant cette génération pour des valeurs décroissantes de β et ce jusqu'à ce que β approche 0. L'acceptation des configurations, pour lesquelles $\Delta C_{ij} > 0$, est effectuée à l'aide d'une génération de nombres aléatoires. C'est-à-dire que la valeur de $\exp(-\Delta C_{ij} / \beta)$ est comparée avec un nombre aléatoire entre 0 et 1; si ce nombre est inférieur à la valeur de l'exponentielle, alors la nouvelle configuration est acceptée, sinon, elle est rejetée. On note que l'application de l'algorithme du recuit avec $\beta = 0$ revient à appliquer l'algorithme d'amélioration itérative puisque la règle d'acceptation ne permet alors que des améliorations.

Nous avons donc montré que, sous certaines conditions sur les probabilités d'acceptation et de perturbation, l'algorithme du recuit simulé converge vers un minimum global avec une probabilité de 1, si pour chaque valeur $\beta(\ell)$ du paramètre de contrôle, la chaîne de Markov correspondante est de longueur infinie et si $\beta(\ell)$ converge éventuellement vers 0 lorsque ℓ tend vers l'infini. Ici, ℓ est l'indice relié aux diminutions de température: on a une valeur de ℓ pour chaque chaîne de Markov homogène.

On a mentionné, au début de la section 4.2, qu'il est possible de réduire les probabilités d'acceptation après chaque réarrangement; dans ce cas le processus d'optimisation n'est plus une suite de chaînes de Markov homogènes, mais plutôt une seule chaîne de Markov non homogène où la valeur de la température $\beta(k)$ est réduite après chaque réarrangement k.

Certains auteurs [4, 11, 12, 14, 25] se sont intéressés à la convergence de l'algorithme du recuit dans le cas non homogène; dans ce cas, le comportement asymptotique est plus délicat à étudier: par exemple, Siarry [30] mentionne l'apparition possible de phénomènes transitoires. Les résultats, que ces auteurs obtiennent, sont basés soit sur des théorèmes d'ergodicité pour chaînes de Markov non homogènes, soit sur l'analogie en temps continue des chaînes de Markov non homogènes. Nous nous contenterons ici d'évoquer le principal résultat de ces travaux d'intérêt théorique: l'algorithme du recuit converge vers un optimum global, avec une probabilité égale à l'unité si, lorsque le nombre de réarrangements k tend vers l'infini, la température $\beta(k)$ ne décroît pas plus vite que l'expression $a/\ln(k)$, où a désigne une constante qui est liée à la profondeur des "puits d'énergie" du problème. De plus, la preuve de convergence asymptotique de l'algorithme non homogène est restreinte au vecteur d'équilibre de forme exponentielle de l'équation (29). Aarts et van Laarhoven [3] affirment qu'il ne semble pas y avoir, dans la littérature, de preuve de convergence pour le cas plus général de l'équation (16).

4.3 Paramètres du programme de recuit.

Dans cette section, nous discuterons de l'implantation de l'algorithme du recuit simulé, en nous restreignant à la règle d'acceptation standard (Éq. 28). Dans la section précédente nous avons montré que l'algorithme du recuit converge asymptotiquement vers une configuration optimale (globale). Cependant, en analysant l'efficacité de l'algorithme, Aarts et van Laarhoven [1] ont montré que sa convergence vers un optimum global peut nécessiter un nombre de réarrangements proportionnel à une fonction exponentielle de la dimension du problème; ce qui entraîne des temps d'exécution exponentiels. Siarry [30] mentionne qu'on peut montrer que la vitesse de convergence est améliorée lorsque l'on part d'une température élevée et que l'on fait décroître celle-ci par paliers. Cette procédure nécessite l'utilisation d'un programme de recuit, qui définit les valeurs optimales des paramètres de la descente en température. On peut même obtenir des solutions quasi optimales, avec des temps polynomiaux, en choisissant des valeurs appropriées pour les paramètres qui contrôlent la convergence. Nous allons examiner successivement les quatre principaux paramètres, à savoir:

- La température initiale $\beta(0)$,

- la longueur des chaînes de Markov homogènes $L(\ell)$, c'est-à-dire le critère de changement de paliers de température,

- la loi de décroissance de la température f $(\beta(\ell + 1) = f(\beta(\ell)))$,

- le critère d'arrêt du programme.

La détermination de paramètres adéquats, pour obtenir un programme de recuit efficace, est devenue un domaine de recherche important. La littérature nous donne bon nombre de programmes de conception simple qui sont similaires au programme original introduit par Kirkpatrick et al. [18]: l'algorithme débute avec une valeur de $\beta(0)$ déterminée de façon expérimentale pour laquelle le rapport d'acceptation $A(\beta(0))$ est près, mais inférieur à 1 ($A(\beta(\ell))$ = le nombre de réarrangements acceptés/nombre de réarrangements proposés à la température $\beta(\ell)$). En second lieu, une suite de chaînes de Markov est générée (chacune à une valeur déterminée de $\beta(\ell)$) pour des valeurs décroissantes de $\beta(\ell)$, où $\beta(\ell + 1) = \alpha\, \beta(\ell)$ (α est une constante près de 1). La longueur de chaque chaîne est déterminée par un nombre minimal de réarrangements acceptés et elle est limitée par un nombre maximal de réarrangements proposés.

Plusieurs auteurs ont proposé des programmes de recuit plus élaborés. Nous allons maintenant étudier différents paramètres qui se retrouvent dans ces programmes.

a) Température initiale:

Il existe une condition nécessaire, mais non suffisante, pour éviter que le processus d'optimisation soit piégé dans un minimum local: la température initale $\beta(0)$ doit être suffisamment élevée pour que, à l'issue du premier palier, toutes les configurations puissent être obtenues avec la même probabilité (voir Éq. (17)). Une expression convenable de $\beta(0)$, qui assure un taux d'acceptation voisin de 1, est la suivante:

$$\beta(0) = t \cdot \max_{i,\, j \in I_R} \Delta C_{ij}, \tag{30}$$

avec $t >> 1$ (typiquement $t = 10$).

En pratique, dans de nombreux problèmes d'optimisation combinatoire, cette règle est d'un emploi malaisé, car il est difficile d'évaluer a priori la différence de coût maximale entre deux configurations quelconques. Le choix de $\beta(0)$ pourra, dans ce cas, résulter d'une procédure expérimentale, effectuée préalablement au processus d'optimisation proprement dit. Au cours d'une telle procédure, on calcule l'évolution du système pendant un temps limité; on acquiert de la sorte une connaissance de l'espace des configurations, à partir de laquelle on détermine $\beta(0)$. Siarry [30] mentionne une expérience préliminaire simple qui consiste à calculer la valeur moyenne de la variation de coût ΔC_{ij}, en maintenant la température à zéro. Aarts et van Laarhoven [1] proposent une procédure plus sophistiquée: ils ont établi une formule itérative qui permet d'ajuster la valeur de $\beta(0)$ après chaque réarrangement préliminaire, de manière à maintenir constant un taux d'acceptation A_0 donné. On choisit initialement $\beta(0) = 0$ et si on effectue m_0 réarrangements préliminaires alors, après chacun, la nouvelle valeur de $\beta(0)$ est donnée par:

$$\beta(0) = \overline{\Delta C^+} \left(\ln \frac{m_2}{m_2 A_0 - (1 - A_0)\, m_1} \right)^{-1}, \tag{31}$$

avec m_1 et m_2 les nombres de réarrangements pour lesquels $\Delta C_{ij} \leq 0$ et $\Delta C_{ij} > 0$ respectivement (à la fin $m_1 + m_2 = m_0$), et $\overline{\Delta C^+}$ la valeur moyenne des $\Delta C_{ij} > 0$. Les auteurs indiquent que cet algorithme conduit à de bons résultats dans le cas où les valeurs de la fonction de coût, pour les diverses configurations du système, sont distribuées de manière suffisamment uniforme. D'autres auteurs [16, 20, 34] ont fourni des expressions pour $\beta(0)$ en se basant sur la différence moyenne de coût entre deux configurations.

b) Longueur des chaînes de Markov et loi de décroissance de la température:

La longueur des chaînes de Markov, qui détermine la durée des paliers de température, et la loi de décroissance de la température, qui joue sur le nombre de paliers, sont deux paramètres du programme de recuit très étroitement liées et sont des plus critiques vis-à-vis du temps de calcul de la méthode. Nous allons introduire le concept de quasi équilibre qui est essentiel pour l'étude de la convergence du recuit simulé.

Soit $L(\ell)$ la longueur d'une chaîne de Markov à la température $\beta(\ell)$ et soit $\mathbf{p}(\beta(\ell))$ le vecteur de distribution des configurations après $L(\ell)$ réarrangements donnés par

$$\mathbf{p}^t(\beta(\ell)) = \mathbf{a}^t \mathbf{P}^{L(\ell)}(\beta(\ell)), \tag{32}$$

Alors on dit que le processus a atteint le quasi équilibre à $\beta(\ell)$ pour $\varepsilon > 0$ si

$$\| \mathbf{p}(\beta(\ell)) - \mathbf{q}(\beta(\ell)) \| \leq \varepsilon . \tag{33}$$

De façon intuitive, il est clair qu'une forte décroissance de $\beta(\ell)$ nécessite plus de réarrangements (chaîne de Markov plus longue) pour rétablir le quasi équilibre à la nouvelle valeur $\beta(\ell+1)$. Il y a donc un juste milieu entre la décroissance rapide de $\beta(\ell)$ et la longueur des chaînes $L(\ell)$. Dans la littérature, on retrouve deux classes de programmes de recuit: ceux qui ont un taux de décroissance de la température constant et une longueur de chaîne variable, et ceux qui ont un taux de décroissance de la température variable et une longueur de chaîne constante. Les programmes de recuit sophistiqués de la première classe sont moins répandus: dans ceux-ci la longueur des chaînes peut être basée sur l'espérance du nombre de réarrangements nécessaires pour s'échapper d'un minimum local. De cette façon, le processus est supposé rester en quasi équilibre durant toute l'optimisation [28] . Il existe évidemment d'autres fonctions pour la longueur des chaînes [10] , mais nous allons

plutôt nous attarder à la deuxième classe qui est beaucoup plus étudiée dans la littérature [10] .

Une première approche du problème consiste à chercher la solution optimale, en fixant la longueur $L(\ell)$ des chaînes de Markov à une valeur donnée M, de manière à atteindre le quasi équilibre pour ε fixé à l'avance. Aarts et van Laarhoven [1] ont obtenu, en utilisant les résultats de la théorie des chaînes de Markov [29] , la condition suivante pour le quasi équilibre:

$$M > D(|R|^2 - 3|R| + 3) \qquad (34)$$

où D est une constante qui dépend de ε . Dans beaucoup de problèmes d'optimisation combinatoire, le nombre total de configurations $|R|$ est une fonction exponentielle du nombre N de variables du problème. Par la suite, l'utilisation de chaînes respectant l'équation (34) conduit à un temps de calcul exponentiel. Ce fait a été confirmé expérimentalement par Aarts et van Laarhoven [1] pour une forme particulière du PVC (les villes occupent toutes les intersections d'un réseau quadrillé plan). De plus, cette expérience montre qu'un gain considérable de temps de calcul est obtenu si l'on accepte de s'écarter un peu de l'optimum: un écart du résultat final de 2%, par rapport à l'optimum, permet de passer d'un temps de calcul exponentiel en fonction de N à un temps cubique. De là l'idée de chercher les paramètres du programme de recuit qui assurent une déviation donnée par rapport à l'optimum, et ce, indépendamment de la dimension du problème considéré.

Le postulat de départ du raisonnement est le suivant: pour chaque chaîne de Markov homogène engendrée au cours du processus d'optimisation, la distribution des états doit être proche de la distribution stationnaire (c'est-à-dire la distribution de Boltzmann (Éq. (29)), si l'on adopte les règles d'acceptation de Metropolis (Éq. (28))).

Cette situation peut être réalisée en partant d'une température élevée (pour laquelle on arrive rapidement au quasi équilibre, comme l'indique la

propriété 3 (Éq. (17))). Puis, il faut choisir le taux de décroissance de la température de manière telle que les distributions stationnaires correspondant à deux valeurs successives de $\beta(\ell)$ soient voisines. De la sorte, après chaque changement de palier de température, la distribution des états s'approche rapidement de la nouvelle distribution stationnaire, si bien que la longueur des chaînes successives peut être maintenue petite: on voit là l'interaction très forte qui existe entre la longueur des chaînes de Markov et le taux de décroissance de la température.

Soient $\beta(\ell)$ et $\beta(\ell+1)$ les températures de deux paliers successifs quelconques et α le taux de décroissance de la température ($\beta(\ell+1) = \alpha\,\beta(\ell) < \beta(\ell)$). La condition à réaliser s'écrit donc:

$$\| q(\beta(\ell)) - q(\beta(\ell+1)) \| < \varepsilon , \tag{35}$$

ou de façon équivalente (mais plus facile d'utilisation)

$$\forall i \in I_R, \quad \frac{1}{1+\delta} < \frac{q_i(\beta(\ell))}{q_i(\beta(\ell+1))} < 1+\delta \tag{36}$$

(ε et δ sont deux nombres réels positifs et petits). Après quelques approximations et manipulations, Aarts et van Laarhoven [1] obtiennent le taux de décroissance de la température suivant:

$$\alpha = \frac{3\,\sigma(\beta(\ell))}{3\,\sigma(\beta(\ell)) + \beta(\ell)\cdot \ln(1+\delta)} \tag{37}$$

où $\sigma(\beta(\ell))$ est l'écart type des valeurs de la fonction de coût pour les configurations de la chaîne de Markov à la température $\beta(\ell)$. Il existe évidemment d'autres expressions pour le taux de décroissance [10].

Si le quasi équilibre est maintenu durant le processus d'optimisation, la longueur des chaînes peut être gardée petite. Cependant, on peut croire que pour des problèmes de plus grande taille, le rétablissement du quasi

équilibre sera plus long; on doit donc s'assurer que le système a une probabilité suffisamment grande de visiter au moins toutes les configurations qui appartiennent au voisinage R_i d'une configuration i donnée. Aarts et van Laarhoven [1] préconisent donc comme longueur de chaînes la valeur suivante:

$$M = \max_{i \in I_R} \{|R_i|\} \tag{38}$$

Le formalisme des chaînes de Markov conduit ainsi à un programme de recuit caractérisé par une longueur de chaîne constante et un taux de décroissance de la température variable. Il faut noter que ce résultat, qui s'appuie sur la théorie, diffère de l'approche empirique usuelle [2, 7, 8, 18] : dans ce cas, on adopte une longueur de palier de température variable et un taux α de décroissance de la température constant, typiquement compris entre 0,80 et 0,99. On observe alors que le paramètre α n'est pas très critique vis-à-vis la convergence de l'algorithme, sous réserve que le palier de température dure assez longtemps.

c) Critère d'arrêt du programme.

Une information quantitative sur la progression du processus d'optimisation peut être tirée de l'entropie, qui est une mesure naturelle de l'ordre du système (une haute entropie indique le désordre et vice versa). On définit l'entropie par l'expression suivante [33] :

$$S(\beta) = - \sum_{i=1}^{|R|} q_i(\beta) \cdot \ln q_i(\beta). \tag{39}$$

En utilisant la distribution stationnaire de Boltzmann (Éq. (29)), Aarts et van Laarhoven [1] obtiennent la forme suivante pour l'entropie:

$$S(\beta) = S(\beta(1)) - \int_{\beta}^{\beta(1)} \frac{\sigma^2(\beta')}{\beta'^3}\, d\beta' \qquad (40)$$

où $\sigma^2(\beta)$ est estimé numériquement à partir des valeurs de la fonction de coût pour les configurations obtenues à la température β. Un critère d'arrêt peut ensuite être élaboré à partir du rapport suivant, qui mesure l'écart entre la configuration courante et la configuration optimale:

$$\frac{S(\beta) - S_o}{S_\infty - S_o} \qquad (41)$$

où S_∞ et S_o sont définis par les relations suivantes:

$$S_\infty = \lim_{\beta \to \infty} S(\beta) = \ln|R|$$

$$S_o = \lim_{\beta \downarrow 0} S(\beta) = \ln|R_o| \quad . \qquad (42)$$

On peut aussi détecter la transition désordre-ordre (et par suite décider de ralentir le refroidissement) par l'observation d'une augmentation brutale du paramètre suivant, qui est analogue à la chaleur spécifique [18] :

$$\sigma^2(\beta) / \beta^2 \quad . \qquad (43)$$

Pour des raisons de précision du calcul numérique, ces critères ne sont applicables en pratique que lorsque les chaînes de Markov sont de longueur suffisante (entre autres, la longueur donnée par l'équation (38) est insuffisante). Dans le cas contraire, un autre critère d'arrêt peut être obtenu à partir de l'extrapolation, à température nulle, de la moyenne lissée, soit $\overline{C_1(\beta)}$, des valeurs de la fonction de coût obtenues au cours du processus d'optimisation [1] :

$$\left| \frac{d\overline{C}_1(\beta)}{d\beta} \cdot \frac{\beta}{\overline{C}(\beta(0))} \right| < \varepsilon_S \qquad (44)$$

où ε_S est un nombre positif et petit, et $\overline{C}(\beta(0))$ est la valeur moyenne de la fonction de coût à la température initiale. Cette règle est issue du fait qu'à basse température $\beta \cdot \left(d\overline{C}_1(\beta) / d\beta \right)$ est un estimateur de $\overline{\Delta C}(\beta) = \overline{C}_1(\beta) - C(r_{i_0})$ (r_{i_0} est une configuration optimale) et que lorsque la différence $\overline{\Delta C}(\beta)$ est petite par rapport à $C(\beta(0))$, alors le processus d'optimisation est proche d'une solution optimale.

Il existe évidemment d'autres critères d'arrêt moins élaborés ou plus arbitraires; mais l'utilisation de programmes de recuit plus recherchés a permis à Aarts et van Laarhoven [1] et Lundy et Mees [22] de montrer l'existence d'une borne supérieure pour le nombre total de paliers de température. En outre, les premiers auteurs ont obtenu une borne proportionnelle à $\ln|R|$ en utilisant les équations (37) et (44). De plus, si l'on fixe la longueur des chaînes conformément à l'équation (38), le temps d'exécution de l'algorithme du recuit est proportionnel à l'expression suivante:

$$\ln |R| \cdot \max_{i \in I_R} |R_i| . \qquad (45)$$

Or le deuxième facteur de cette équation est le plus souvent une fonction polynomiale du nombre de variables du problème; ce qui fait que l'algorithme du recuit simulé, utilisé avec le programme constitué par les équations (37), (38) et (44), procure un résultat quasi optimal, en un temps polynomial, pour la plupart des problèmes. Nous allons voir, dans la prochaine section, des applications du recuit simulé à des problèmes d'optimisation combinatoire.

5) Applications aux problèmes d'optimisation combinatoire.

Le recuit simulé est applicable à la plupart des problèmes d'optimisation combinatoire et l'algorithme qui en résulte est facile d'implantation. L'application d'un algorithme de recuit peut être divisé en trois étapes distinctes:

a) Formulation d'une représentation concise du problème.

b) Formulation d'un mécanisme de transition.

c) Optimisation en suivant un programme de recuit.

Voyons plus en détails chacune d'elles.

a) Une description concise d'un problème consiste en une représentation des configurations et une expression pour la fonction de coût. Celle-ci doit être définie de telle sorte qu'elle représente bien la "qualité" de ce qu'on cherche à optimiser. Il est préférable d'exprimer la représentation des configurations et la fonction de coût par des expressions simples qui sont faciles à manipuler.

b) La transformation d'une configuration à une autre nécessite trois étapes: premièrement, on doit perturber le système, deuxièment, on doit calculer la différence de coût et troisièmement, on doit prendre la décision d'accepter ou non la nouvelle configuration. La perturbation du système est habituellement faite en générant de façon aléatoire les nouvelles configurations et ce, en utilisant des réarrangements relativement simples (locaux) (par exemple: permutations, inversions, échanges, translations, rotations, etc...). L'évaluation de la différence de coût doit être très simple puisque, en pratique, c'est la partie la plus coûteuse de l'algorithme en temps de calcul. La décision d'accepter une nouvelle configuration est effectuée en appliquant une règle d'acceptation comme celle de l'équation (28).

c) L'optimisation suivant un programme de recuit, implique le choix des paramètres qui déterminent le programme. Par exemple, pour leprogramme utilisant les équations (31), (37) et (44), les paramètres sont le taux d'acceptation initial A_0, le paramètre de distance δ et le paramètre d'arrêt ε_S .

Il est évident que l'habileté d'un utilisateur du recuit simulé réside dans sa façon de manipuler les trois items précédents.

Le recuit simulé a été appliqué à plusieurs problèmes d'optimisation, soit pratiques, soit théoriques dans une grande variété de disciplines [10] . Bon nombre d'auteurs ont fait des recherches sur les performances du recuit simulé en utilisant des cas particuliers de problèmes théoriques (nous verrons quelques cas plus loin). On peut tirer une conclusion générale de ces études: le recuit simulé permet d'obtenir des solutions quasi optimales, mais certains problèmes nécessitent de grands efforts de calcul. Pour des cas particuliers générés aléatoirement, les performances du recuit simulé sont excellentes. Il faut mentionner que pour la plupart des problèmes d'optimisation théoriques étudiés, il existe des algorithmes spécialisés [19, 26] . Malheureusement, il n'existe pas vraiment, en ce moment, d'étude sur la comparaison entre le recuit simulé et ces algorithmes spécialisés; il faut procéder article par article et essayer de trouver des points de comparaison. Il n'y a que l'article (non publié) de Johnson et al [16] qui présente une telle comparaison pour quatre types de problèmes. Dans cet article, le recuit simulé surclasse l'algorithme spécialisé pour le problème du partitionnement de graphes, mais il est surclassé pour les problèmes de voyageur de commerce, de coloriage de graphes ainsi qu'un autre type de problème de partitionnement. Les auteurs concluent, de plus, que le recuit simulé est plutôt lent, ce qui est aussi affirmé par d'autres auteurs. Mais des conclusions de ce type ne nous indiquent pas grand-chose quant à la qualité du recuit simulé, puisque les comparaisons sont faites à l'aide d'un seul programme de recuit; rien ne nous assure que le programme qu'ils ont utilisé est le plus efficace pour le problème étudié. On peut cependant tirer une conclusion fort importante: le recuit simulé n'est pas une recette miracle. Ceci veut simplement dire

que pour développer un algorithme efficace, en se basant sur le recuit, il faut faire des efforts: chaque problème particulier nécessite un réglage minutieux. Nous allons voir maintenant des exemples d'application pour différents problèmes d'optimisation combinatoire et constater la diversité des résultats rencontrés, tant du point de vue des programmes de recuit utilisés que de celui de la qualtité des solutions obtenues.

5.1 Problème du voyageur de commerce.

On rappelle la définition du problème du voyageur de commerce (PVC): on doit visiter N villes, une et une seule fois, revenir au point de départ et ce en minimisant la distance parcourue. Pour appliquer l'algorithme, on doit passer par les trois étapes mentionnées plus haut. Premièrement, nous allons voir les résultats obtenus par Aarts et van Laarhoven [1] sur un cas particulier théorique: les villes occupent toutes les intersections d'un quadrillage plan. Une configuration est représentée par le vecteur $r_i = \left(r_{i_1}, r_{i_2}, \ldots, r_{i_{N+1}} \right)$, où $r_{i_1} = r_{i_{N+1}}$ et $r_{i_j} \in [1, N] \subset \mathbb{N}$ est le numéro de la j^{e} ville de l'itinéraire i. La fonction de coût est la distance moyenne entre deux villes et est donnée par

$$C(r_i) = \sum_{\ell = 1}^{N} d(r_{i_\ell}, r_{i_{\ell+1}})/N \qquad (46)$$

où $d(r_{i_\ell}, r_{i_{\ell+1}})$ dénote la distance entre deux villes successives et celle-ci est définie par:

$$d(r_{i_\ell}, r_{i_{\ell+1}}) = |x_{i_{\ell+1}} - x_{i_\ell}| + |y_{i_{\ell+1}} - y_{i_\ell}|, \qquad (47)$$

où (x_k, y_k) sont les coordonnées de la $k^{\underline{e}}$ ville dans notre quadrillage plan. Cette distance est connue sous le nom de distance de Manhattan. Les réarrangements sont effectués en choisissant au hasard des mouvements 2-opt tels que définis par Lin et Kernighan [21]: on choisit deux villes au hasard dans r_i et on inverse l'ordre de visite, dans l'itinéraire, de toutes les villes comprises entre les deux villes choisies, c'est-à-dire si on choisit la $m^{\underline{e}}$ et la $n^{\underline{e}}$ ville dans l'itinéraire $(n > m)$,

$$\begin{aligned} r_{j_\ell} &= r_{i_\ell} && \text{si} \quad 1 \le \ell \le m \ \text{ou} \quad n \le \ell \le N, \\ r_{j_\ell} &= r_{i_{n+m-\ell}} && \text{si} \quad m+1 \le \ell \le n-1 . \end{aligned} \tag{48}$$

Ceci est équivalent, de façon plus intuitive, à enlever deux liens dans l'itinéraire et à les remplacer par deux nouveaux, tout en respectant les contraintes. La différence de coût est très facile à calculer pour ce type de réarrangement, car elle implique seulement quatre opérations élémentaires.

Le programme de recuit utilisé est donné par les équations (30) et (34) pour la température initiale $(\beta(0))$ et la longueur des chaînes (M); les auteurs utilisent un taux de décroissance de la température constant (α). Pour ce cas particulier du PVC, la solution optimale est connue et le but des auteurs était d'étudier le comportement de l'algorithme lorsque le nombre de villes augmente; ils ont donc utilisé un programme de recuit très simple: la température initiale $\beta(0) = 1,7$ est donnée par l'équation (30), le taux de décroissance de la température est constant $\alpha = 0,95$. L'algorithme commence avec une longueur de chaînes restreinte et fonctionne tant qu'on n'a pas obtenu une configuration qui demeure inchangée pas pour 10 chaînes subséquentes; si la solution n'est pas optimale ou quasi optimale (selon l'objectif recherché) on augmente la longueur des chaînes par un facteur de 1,2 et on recommence le processus d'optimisation. De cette façon, les auteurs ont étudié la longueur des chaînes, nécessaires pour une taille de problème donnée, ainsi que le temps nécessaire pour obtenir les résultats escomptés. Les

auteurs obtiennent des temps de calcul exponentiels pour les solutions optimales et cubiques pour les solutions quasi optimales; ils affirment que ces résultats sont similaires à ceux obtenus par Lin et Kernighan [21] lorsque ces derniers utilisent des réarrangements plus complexes (par exemple 3-opt, où on enlève 3 liens plutôt que 2). De plus, Aarts et van Laarhoven montrent que le critère d'arrêt basé sur l'entropie (Éq. (41)) est peu pratique puisqu'il nécessite (pour être fiable) des chaînes de Markov beaucoup plus longues qu'il est nécessaire pour obtenir de bonnes solutions.

Dans le même article, ces auteurs utilisent un programme plus intéressant défini par les équations (37), (38) et (44). La longueur des chaînes de Markov est fixée à $M = N(N-1)/2$, le paramètre de distance pour le taux de décroissance de la température est fixé à $\delta = 0,1$, le paramètre pour le critère d'arrêt à $\varepsilon_S = 0,001$. Le problème particulier qu'ils étudient comporte 225 villes. Les résultats obtenus indiquent que les paramètres δ et ε_S sont appropriés pour prévoir le résultat final du processus d'optimisation (plus δ et ε_S sont petits, meilleure est la solution). Les résultats obtenus sont quasi optimaux (écart de 2%) et supérieurs à l'algorithme d'amélioration itérative (écart de 7%), et les temps de calcul sont comparables à ceux obtenus pour le programme précédent (cubique).

5.2 Problème d'affectation quadratique.

Les problèmes d'affectation quadratique (PAQ) sont des problèmes d'optimisation combinatoire qu'on représente à l'aide d'un ensemble $N = \{1, 2, ..., n\}$ et de deux matrices $(n \times n)$ $A=(a_{ij})$ et $B = (b_{kl})$ composées d'éléments réels donnés. La matrice A est appelée la matrice des distances et la matrice B, la matrice de connexion. Il faut déterminer une permutation φ de l'ensemble N telle que la somme

$$\sum_{i \in N} \sum_{j \in N} a_{ij} \; b_{\varphi(i)\varphi(j)} \qquad (49)$$

est minimale.

Nous allons examiner ici les résultats obtenus par Burkard et Rendl [8] pour le problème d'affectation quadratique. Dans celui-ci une configuration est représentée par une permutation φ de l'ensemble N et la fonction de coût est définie par l'équation (49). Les réarrangements sont effectués en choisissant au hasard deux éléments de la permutation et en les interchangeant de position. La variation de coût est facile à calculer: elle implique un nombre d'opérations de l'ordre de n.

Le programme de recuit utilisé est très simple: on choisit arbitrairement tous les paramètres: la température initiale, le taux de décroissance de la température et la longueur des chaînes. Il n'y a pas à proprement parlé de critère d'arrêt: l'algorithme se termine lorsqu'il n'est plus capable d'accepter de changement et ceci se produit après un nombre suffisant d'itérations. En effet, à ce moment, la température étant très basse, la règle d'acceptation ne permet plus de détérioration et devient équivalente à l'algorithme d'amélioration itérative. Malgré ce programme très simple, les auteurs rapportent de bons résultats: l'algorithme de recuit serait comparable, tant du point de vue qualité de solution que de celui du temps de calcul, aux autres algorithmes sophistiqués et réputés rapides.

5.3 Problème de localisation p-médiane.

Le problème de localisation p-médiane est le suivant: il faut choisir p sites parmi les N noeuds d'un réseau R, de telle sorte que la somme des distances de chaque noeud au plus proche site est minimale. R = (G,D) où G = { 1, 2, ..., N } est l'ensemble des noeuds et $D = (d_{ij})$, la matrice (n x n) des distances.

Nous allons examiner les résultats obtenus par Golden et Skiscim [13] pour ce problème. Une configuration est représentée par le sous-ensemble $V_m = \{v_1^m, v_2^m, \ldots, v_p^m\} \subset \{1, 2, \ldots, N\}$; ce sont les p sites choisis. La fonction de coût est la suivante:

$$C(V_m) = \sum_{i=1}^{N} \min_j d_{i v_j^m} \quad . \tag{50}$$

Les réarrangements sont effectués en choisissant au hasard un site qui n'est pas dans la configuration étudiée et en le substituant à un site qui est dans la configuration. L'algorithme choisit le site sortant comme étant celui qui fait diminuer le plus (ou augmenter le moins) le coût; la variation de celui-ci est très facile à calculer. On réfère le lecteur à l'article de Teitz et Bart [32] pour les détails du calcul de la variation de coût.

Le programme de recuit de Golden et Skiscim est du même genre que celui de Burkard et Rendl: très simple et plutôt empirique. La température initiale est choisie arbitrairement (élevée), le nombre de paliers de température est fixé à l'avance et ceux-ci sont espacés uniformément entre 0 et la température initiale, ou bien ils sont diminués par un facteur de 0,90 à chaque changement. Ici encore, il n'y a pas vraiment de critère d'arrêt; on doit plutôt atteindre un certain équilibre contrôlé par un paramètre de distance $\delta = 0,90$. La longueur des chaînes est variable et déterminée par ce même équilibre. Pour vérifier si on a atteint l'équilibre, l'algorithme effectue n réarrangements (n fixé au départ); ce qui constitue une époque, et on vérifie si le coût obtenu est proche (d'où le paramètre δ) d'un coût obtenu pour une époque précédente (au même palier de température): si oui, l'équilibre est atteint; si non, on effectue n autres réarrangements à la même température.

Les auteurs obtiennent des résultats peu encourageants: les temps de calcul et la qualité des solutions obtenues sont surpassés par les heuristiques spécialisées (même celles qui sont peu sophistiquées). De plus, les auteurs affirment que les temps de calcul croissent plus que cubiquement en pratique, et que la qualité des solutions ainsi que les temps de calcul sont très sensibles

aux paramètres de contrôle. Les auteurs obtiennent les mêmes résultats décevants pour le problème du voyageur de commerce.

5.4 Problème du coloriage de graphes.

Le problème du coloriage de graphes est le suivant: soïent un graphe $G = (V,E)$ où V est l'ensemble des noeuds et E l'ensemble des liens et k un entier positif; on doit trouver un k-coloriage de G, c'est-à-dire une partition de V en k ensembles indépendants $V_1, ..., V_k$ (un ensemble V_i est indépendant s'il n'y a pas de liens entre deux noeuds de V_i). Si une telle partition existe, G est dit k-coloriable. Il existe un autre problème très apparenté, c'est celui qui consiste à trouver la plus petite valeur de k telle que G est k-coloriable; mais nous allons nous concentrer sur le premier type de problème.

Pour ce problème, nous étudierons les résultats obtenus par Chams et al. [9]. Dans leur article, une configuration est représentée par la partition $S = (V_1, ..., V_k)$ de l'ensemble V en k sous-ensembles $V_1, ..., V_k$. La fonction de coût est donnée par:

$$f(S) = f(V_1, ..., V_k) = \sum_{i=1}^{k} |E_i| \qquad (51)$$

où $|E_i|$ est le nombre de liens qui ont leurs deux extrémités dans V_i . Nous aurons donc un k-coloriage lorsque $f(S) = 0$. Les réarrangements sont effectués en choisissant deux indices i et j et en déplaçant de V_i à V_j un noeud $x \in V_i$ adjacent à au moins un autre noeud de V_i. Les variations de coût sont très faciles à calculer, puisqu'elles n'impliquent que quelques opérations élémentaires pour les quantités $|E_i|$ et $|E_j|$.

Dans cet article, les auteurs ont utilisé le recuit simulé de deux façons: premièrement, ils ont appliqué la méthode de façon standard en utilisant le même programme de recuit que Burkard et Rendl [8], et deuxièmement, ils ont combiné le recuit à une heuristique développée antérieurement pour le problème du coloriage de graphe. Cette heuristique construit les sous-ensembles indépendants V_i de façon séquentielle de sorte que lorsqu'il reste q noeuds à colorier (q est un paramètre de la méthode combinée), on passe de l'heuristique au recuit simulé. En fait, l'heuristique détermine $p \leq k$ sous-ensembles indépendants et le recuit cherche à placer les q noeuds restants dans $k - p$ sous-ensembles.

Les résultats obtenus sont plutôt positifs: le recuit est extrêmement efficace pour les graphes de moins de 200 noeuds; pour les graphes de plus grande taille (300 à 1000 noeuds), le recuit est qualifié de bonne heuristique. Cependant, les meilleurs résultats, tant du point de vue efficacité que qualité, sont obtenus avec la méthode combinée et ceux-ci surpassent les résultats de l'heuristique et du recuit lorsqu'ils sont employés seuls.

5.5 Problème de couplage.

Le problème que nous allons étudier est celui du couplage pondéré euclidien de poids minimal. Soient G un ensemble de N points et $D = (d_{ij})$, une matrice (N x N) d'éléments réels représentant les distances euclidiennes entre les points. Un couplage parfait de G est un ensemble de $N/2$ liens tels que chaque point de G est une extrémité de un et un seul des $N/2$ liens. Le problème est de trouver un couplage parfait tel que la somme des distances (ou poids) des $N/2$ liens est minimale.

Nous examinons, dans ce qui suit, les résultats qu'ont obtenus Lutton et Bonomi [23] pour le problème de couplage pondéré à poids minimal. Chaque configuration est représentée par un vecteur π_i

$= ((b^i_1, e^i_1), (b^i_2, e^i_2), \ldots, (b^i_{N/2}, e^i_{N/2}))$ où chaque couple représente un lien d'extrémité initiale b^i_j et d'extrémité terminale e^i_j. La fonction de coût utilisée est la suitvante:

$$C(\pi_i) = \sum_{j=1}^{N/2} d\, b^i_j e^i_j \,. \qquad (52)$$

Avant d'étudier le type de réarrangements utilisés par ces auteurs, on doit mentionner que ceux-ci ont partitionné la région qui contient les points, en sous-régions, et ceci dans le but de limiter la longueur des liens. L'idée sous-jacente est que, pour minimiser la distance, on doit relier des points qui sont relativement proches dans le plan euclidien. La configuration initiale est donc déterminée en respectant cette idée: chaque point est lié à un point de sa sous-région ou d'une sous-région adjacente. Les réarrangements sont alors effectués en choisissant au hasard des mouvements de type 2-opt comme dans le cas du problème de voyageur de commerce: on enlève deux liens dans le couplage, (i_1, i_2) et (j_1, j_2), qu'on remplace par les liens (i_1, j_1) et (i_2, j_2). La variation de coût est évidemment très facile à calculer. Pour choisir les deux liens, les auteurs suggèrent la méthode suivante: on choisit au hasard une sous-région S_k, puis encore au hasard, un point i_1 de cette sous-région. Puisqu'il est nécessairement lié à un autre point i_2, on a alors choisi de façon aléatoire le premier lien (i_1, i_2). On choisit ensuite au hasard un point j_1 dans la sous-région S_k ou dans l'une de ses sous-régions adjacentes et on obtient le deuxième lien (j_1, j_2). Le choix de ce second lien est effectué de la sorte dans le but de respecter l'idée de relier des points qui sont proches dans le plan euclidien. On réfère le lecteur au texte des auteurs, pour la façon de partitionner la région de départ.

Le programme de recuit utilisé par Lutton et Bonomi est simple, mais possède une particularité encore jamais rencontrée: la température initiale est basse. En effet, de par la structure euclidienne du problème, les auteurs utilisent une configuration initiale et des réarrangements qui correspondent toujours à des basses températures; c'est-à-dire qu'on ne visite jamais des

configurations de coût élevé et donc qu'on n'aura jamais besoin de température très élevée pour sortir d'un minimum local. En résumé, la température initiale est fixée arbitrairement à une valeur peu élevée, le taux de décroissance est constant, le nombre de chaînes est fixé à l'avance et la longueur de celles-ci a été déterminée de façon empirique selon la taille du problème.

Le problème de couplage pondéré euclidien de poids minimal n'est habituellement pas considéré comme un problème difficile, puisqu'il existe un algorithme qui le résout de façon exacte avec un temps qui est une fonction cubique du nombre de points N. Cependant, lorsque N devient grand (N > 1000), même cette méthode polynomiale nécessite un temps exhorbitant; c'est pourquoi il faut développer des heuristiques plus rapides [5,15]. Mais en étant plus efficaces ces dernières fournissent des solutions de 30% à 50% plus coûteuses que la solution optimale. L'algorithme de recuit, élaboré par Lutton et Bonomi, est moins rapide que ces algorithmes, mais fournit des solutions à peine plus coûteuses (faible pourcentage d'augmentation) que la solution optimale dans des temps raisonnables. Les auteurs affirment que ceci fait du recuit simulé une des meilleures méthodes pour ce type de problème.

5.6 Partitionnement de graphes.

Le problème de partionnement de graphes s'énonce comme suit: on veut partitionner les noeuds d'un graphe en deux sous-ensembles ou plus, de telle sorte que le nombre total de liens qui passent d'un sous-ensemble à un autre soit minimal. On exige aussi que chaque sous-ensemble ait le plus possible la même cardinalité. Le problème particulier que l'on va étudier est représenté par le graphe $G = (V,E)$, où $V = \{V_i\}$ est l'ensemble des noeuds et $E = \{e_j\}$ est l'ensemble des liens. Soient les ensembles $V^-, V^+ \subset V$, tels que $V = V^- \cup V^+$ et $V^- \cap V^+ = \phi$, et l'ensemble $E^o = \{e = (v_{i_1}, v_j) \in E | v_i \in V^- \text{ et } v_j \in V^+\}$, alors le problème est de construire les ensembles V^- et V^+ de telle sorte que $\left| |V^-| - |V^+| \right|$ et $|E^o|$ soient minimaux.

Nous allons examiner les résultats obtenus par Aarts et van Laarhoven [2] pour ce problème particulier. Une configuration est représentée par le vecteur r^i défini par:

$$r^i_j = \begin{cases} -1 & \text{si} \quad v_j \in V^- \\ 1 & \text{si} \quad v_j \in V^+ \end{cases} \tag{53}$$

Chaque vecteur r_i correspond à une partition de V. La fonction de coût utilisée est la suivante:

$$C(r^i) = \sum_{j=1}^{|V|-1} \sum_{k=j+1}^{|V|} (\lambda - a_{jk}/2)\ r^i_j\ r^i_k , \tag{54}$$

où a_{jk} est le nombre de liens entre les noeuds v_j et v_k, et λ, un facteur de pondération égal à $\bar{a}/2$, ($\bar{a}$ est le nombre moyen de liens par noeud). Les termes a_{jk} sont associés à la minimisation de $|E^o|$ et le facteur λ à la minimisation de $\left| |V^-| - |V^+| \right|$. Les réarrangements sont effectués en choisissant au hasard un noeud et en le changeant de sous-ensemble. La variation de coût est effectuée à l'aide d'une sommation du type de l'équation (54) mais comportant un seul indice de variation.

Les auteurs ont comparé trois programmes de recuit différents: le premier est déterminé, comme c'était le cas pour le deuxième programme de recuit appliqué au PVC, par les équations (31), (37), (38) et (44). Les paramètres choisis sont: le taux d'acceptation pour la température initiale $A(0) = 0{,}95$, le paramètre de distance pour le taux de décroissance $\delta = 0{,}1$, le critère d'arrêt $\varepsilon_S = 0{,}001$ et la longueur des chaînes $M = |V|$. Le deuxième programme utilise la même température initiale et la même longueur de chaînes, mais le taux de décroissance de la température est donné par une fonction du taux d'acceptation et le critère d'arrêt est le suivant: on arrête le programme si après 4 chaînes consécutives il n'y a pas eu d'amélioration. Le

troisième programme utilise encore la même température initiale, un taux de décroissance fixe (α = 0,95), une longueur de chaîne variable, limitée par un taux minimum d'acceptation et un nombre maximum de réarrangements, et le critère d'arrêt est le même que pour le deuxième programme.

Les résultats obtenus pour ces trois programmes ont été comparés à ceux de la méthode d'amélioration itérative. Les auteurs parviennent à améliorer la qualité de la solution avec chacun des programmes, mais le plus intéressant est sans nul doute le premier; il est plus rapide et fournit une meilleure solution que les deux autres. On note une amélioration de la solution de près de 15% mais avec un temps de calcul d'environ 15 fois supérieur.

6) Conclusion.

Comme l'avaient suggéré ou prédit Kirkpatrick et al [18], l'analogie entre la physique statistique et l'optimisation combinatoire s'est avérée plutôt significative. En effet, l'application de la méthode du recuit simulé aux problèmes d'optimisation nous assure d'obtenir de façon théorique des solutions optimales et en pratique, des solutions quasi optimales. L'algorithme de recuit peut être vu comme un algorithme général d'optimisation; c'est-à-dire que le même schéma peut être utilisé pour tous les types de problèmes combinatoires, mais il faut cependant en ajuster les composantes pour chaque type de problème.

Les avantages de l'algorithme de recuit simulé sont les suivants: on peut obtenir des résultats quasi optimaux, l'algorithme peut être exécuté dans des temps polynomiaux et il est généralement applicable et facile à implanter. L'inconvénient majeur de l'algorithme est qu'il nécessite, pour certaines applications, beaucoup de temps de calcul. De plus, pour certains types de problèmes, il existe déjà des heuristiques valables; donc on serait porté à croire que l'avenir de l'algorithme de recuit réside dans son application à des domaines où il n'y a pas de telles méthodes disponibles. Cependant, on a vu qu'il est possible de combiner des heuristiques déjà existantes avec le recuit

pour obtenir des algorithmes encore plus efficaces. On a constaté aussi qu'il était possible de modifier l'algorithme de recuit pour tenir compte des connaissances qu'on a déjà acquises sur le problème particulier qu'on étudie.

Plus spécifiquement, on a remarqué que les composantes de l'algorithme de recuit (réarrangements, programme de recuit, etc...), sont très importantes. Certains auteurs ont obtenu des résultats très encourageants et d'autres, des conclusions beaucoup moins positives. Parfois, des programmes de recuit très simples se sont avérés très satisfaisants, parfois les auteurs ont dû user d'astuce et de finesse pour arriver à des solutions adéquates.

En résumé, les chercheurs croient que l'algorithme du recuit simulé peut devenir une technique qui pourra être appliquée à un large éventail de problèmes d'optimisation combinatoire, mais certains restent sceptiques en constatant qu'il faut parfois des temps de calcul assez considérables. L'application de la technique du recuit simulé aux problèmes combinatoires étant très récente, il y a encore beaucoup de recherches à faire; entre autres, celles sur les problèmes à grande échelle et celles sur les façons d'accélérer l'algorithme par exécution parallèle pourraient aboutir à des conclusions importantes sur l'utilité, en pratique, du recuit simulé.

BIBLIOGRAPHIE

[1] Aarts E. H. L. et van Laarhoven P. J. M., *Statistical Cooling: A General approach to Combinatorial Optimization Problems*, Philips Journal of Research, Vol. 40 (1985), No. 4, pp. 193-226.

[2] Aarts E. H. L. et van Laarhoven P. J. M., *A New Polynomial-Time Cooling Schedule*, Proceedings IEEE International Conference on CAD, ICCAD-85, Santa Clara, CA, Novembre 1985, pp. 206-208.

[3] Aarts E. H. L. et van Laarhoven P. J. M., *A Pedestrian Review of the Theory and Application of the Simulated Annealing Algorithm*, Proceedings Heidelberg Colloquium on Glassy Dynamics, Lecture Notes in Physics, Vol. 275 (1987), pp. 288-396.

[4] Anily S. et Federgruen A., *Ergodicity in Parametric Non-Stationary Markov Chains: An Application to Simulated Annealing Methods*, Preprint (1985), Graduate School of Business, Columbia University, New York, NY 10027, to appear in Operations Research.

[5] Avis D., *A Survey of Heuristics for the Weighted Matching Problem*, Network, Vol. 13 (1983), No. 4, pp. 475-493.

[6] Binder K. et Young A. P., *Spin-Glass - Experimental Facts, Theoretical Concepts and Open Questions*, Review of Modern Physics, Vol. 58 (1986), No. 4, pp. 801-976.

[7] Bonomi E. et Lutton J. L., *The N-City Travelling Salesman Problem, Statistical Mechanics and the Metropolis Algorithm*, SIAM Review, Vol. 26 (1984), No. 4, pp. 551-568.

[8] Burkard R. E. et Rendl F., *A Thermodinamica'ly Motivated Simulation Procedure for Combinatorial Optimization Problems*, European Journal of Operational Research, Vol. 17 (1984), No. 2, pp. 169-174.

[9] Chams M., Hertz A. et de Werra D., *Some Experiments with Simulated Annealing for Coloring Graphs*, European Journal of Operational Research, Vol. 32 (1987), pp. 260-266.

[10] Collins N. E., Eglese R. W. et Golden B. L., *Simulated Annealing - An Annoted Bibliography*, Working Paper Series MS/S 87-028, University of Maryland at College Park 20742, Octobre 1987.

[11] Geman S. et Geman D., *Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 6 (1984), No. 6, pp. 721-741.

[12] Gidas B., *Nonstationary Markov Chains and Convergence of the Annealing Algorithm*, Journal of Statistical Physics, Vol. 39 (1985), Nos 1-2, pp. 77-131.

[13] Golden B. L. et Skiscim C. C., *Using Simulated Annealing to Solve Routing and Location Problems*, Naval Research Logistics Quarterly, Vol. 33 (1986), No. 2, pp. 261-279.

[14] Hajek B., *Cooling Schedules for Optimal Annealing*, Report (1986), Department of Electrical Engineering, University of Illinois at Champaign-Urbana, IL 61081, submitted to Mathematics of Operations Research.

[15] Iri M., Murota K. et Matsui S., *Heuristic for Planar Minimum-Weight Perfect Matching*, Network, Vol. 13 (1983), pp. 67-92.

[16] Johnson D. S., Aragon C. R., Mc Geoch L. A. et Schevon C., *Optimization by Simulated Annealing: An Experimental Evaluation (Part I and II)*, Preprint (1987), AT & T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ, 079974-2070.

[17] Kelly F. P., *Reversibility and Stochastic Networks*, Wiley, Chichester, 1979.

[18] Kirkpatrick S., Gelatt C. D. Jr. et Vecchi M. P., *Optimization by Simulated Annealing* , Science, Vol. 220 (1983), No. 4598, pp. 671-680.

[19] Lawler E. L., Lenstra J. K., Rinnoy Kan A. H. G. et Shmoys D. B., *The Traveling Salesman Problem*, Wiley, Chichester, 1985.

[20] Leong H. W., Wong D. F. et Liu C. L., *A Simulated Annealing Channel Router*, Proceedings IEEE International Conference on CAD, ICCAD-85, Santa Clara, CA, Novembre 1985.

[21] Lin S. et Kernighan B. W., *An Effective Heuristic Algorithm for the Travelling Salesman Problem*, Operations Research, Vol. 21 (1973), pp. 498-516.

[22] Lundy M. et Mees A., *Convergence of an Annealing Algorithm*, Mathematical Programming, Vol. 34 (1986), pp. 111-124.

[23] Lutton J. L. et Bonomi E., *Simulated Annealing Algorithm for the Minimum Weighted Perfect Euclidean Matching Problem*, RAIRO Recherche Opérationnelle, Vol. 20 (1986), No. 3, pp. 177-197.

[24] Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H. et Teller E., *Equation of State Calculation by Fast Computing Machines*, Journal of Chemical Physics, Vol. 21 (1953), No. 6, pp. 1087-1091.

[25] Mitra D., Romeo F. et Sangiovanni-Vincentelli A., *Convergence and Finite-Time Behavior of Simulated Annealing*, Advances in Applied Probability, Vol. 18 (1986), No. 3, pp. 747-771.

[26] Papadimitriou C. H. et Steiglitz K., *Combinatorial Optimization : Algorithms and Complexity*, Prentice Hall, New York, 1982.

[27] Rammal R., Toulouse G. et Viraso M. A., *Ultrametricity for Physicists*, Review of Modern Physics, Vol. 58 (1986), No. 3, pp. 765-788.

[28] Romeo F. et Sangiovanni-Vincentelli A., *Probabilistic Hill Climbing Algorithms: Properties and Applications*, Proceedings 1985 Chapel Hill Conference on VLSI, Chapel Hill, NC, 1985, ed. Fuchs H., pp. 393-418.

[29] Seneta E., *Non Negative Matrices and Markov Chains*, Springer Verlag, New York, 2^{e} édition, 1981.

[30] Siarry P., *La méthode du recuit simulé: application à la conception de circuits électroniques*, These de Ph. D., Université de Paris 6, Paris, France.

[31] Solla S. A., Sorkin G. B. et White S. R., *Configuration Space Analysis for Optimization Problems*, NATO ASI Series, Vol. F20 (1986), Disordered Systems and Biological Organization, éditeurs Bienenstock E., Fogelman Soulie F. et Weisbuch G., Springer Verlag, pp. 283-294.

[32] Teitz M. B. et Bart P., *Heuristic Methods for Estimating the Generalized Vertex Median of a Weighted Graph*, Operations Research, Vol. 16 (1968), pp. 955-961.

[33] Toda M., Kubo R. et Saito N., *Statistical Physics*, Springer Series in Solid-State Sciences, Springer, Berlin, 1983.

[34] White S. R., *Concepts of Scale in Simulated Annealing*, Proceeding IEEE Conference on Computer Design: VLSI in Computers, ICCD-84, Port Chester, NY, Octobre 1984, pp. 646-651.